BSTZ No. 042390.P12271
Express Mail No.: EL802874604

UNITED STATES PATENT APPLICATION

FOR

ADAPTIVE FILTERING WITH TAP LEAKAGE USING ERROR FILTERING

INVENTORS:

Stanley K. Ling
Hiroshi Takatori

PREPARED BY:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 Wilshire Blvd., 7th Floor
Los Angeles, CA 90025-1026
(714) 557-3800

BACKGROUND

1. Field of the Invention

This invention relates to adaptive filtering. In particular, the invention relates to adaptive filtering with tap leakage.

2. Description of Related Art

Adaptive filters may be used in a number of applications such as adaptive equalization in digital communications systems. Adaptive filters are typically implemented as a non-recursive filter having N taps. For example, the adaptive filter may be a fractionally spaced equalizer having tap weights spaced a fraction of a symbol. The fractional spaced equalizer provides better performance than conventional symbol synchronous equalizers in the presence of severe linear distortion. The adaptive filter may have N equalizer coefficients that are updated or adjusted according to the input signal. The adaptive filter prevents the buildup of large coefficient values by systematically adjusting the magnitudes of all the equalizer tap weights or equalizer coefficients. However the systematic adjustment may cause output overflow, leading to substantially degraded performance.

A technique to solve this problem includes preventing the buildup of large coefficient values through systematically "leaking" or decreasing the magnitudes of the equalizer tap weights. This technique has a number of disadvantages. First, it requires an additional multiplication for each tap-update. The number of additional operations becomes significant for large number of taps. Second, adding tap-leakage computations to pre-existing hardware blocks may be difficult, considering the number and complexity of additional required operations.

Therefore, there is a need to have an efficient technique to reduce the computations of the adaptive filter with tap leakage.

BRIEF DESCRIPTION OF THE DRAWINGS

The features and advantages of the present invention will become apparent from the following detailed description of the present invention in which:

Figure 1 is a diagram illustrating a system in which one embodiment of the invention can be practiced.

Figure 2 is a diagram illustrating an adaptive filter shown in Figure 1 according to one embodiment of the invention.

Figure 3 is a diagram illustrating a tap-leakage generator shown in Figure 1 according to one embodiment of the invention.

Figure 4 is a diagram illustrating an error filter shown in Figure 3 according to one embodiment of the invention.

Figure 5 is a flowchart illustrating a process for adaptive filtering with tap leakage using error filtering according to one embodiment of the invention.

DESCRIPTION

The present invention is a technique to implement a tap-leakage computation for adaptive filters. The complexity of the tap-leakage computation is reduced from N operations, where N is the number of taps in the adaptive filter, to a computation which is independent of filter length. In one embodiment of the invention, a tap-leakage generator has an error filter and an updater. The error filter is coupled to an adaptive filter having N taps to filter a decision error provided by the adaptive filter using a leakage factor. The updater updates N equalizer coefficients to the N taps using the filtered decision error. The updater receives the N equalizer data from the N taps.

In the following description, for purposes of explanation, numerous details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that these specific details are not required in order to practice the present invention. In other instances, well-known electrical structures and circuits are shown in block diagram form in order not to obscure the present invention.

The present invention may be implemented by hardware, software, firmware, microcode, or any combination thereof. When implemented in software, firmware, or microcode, the elements of the present invention are the program code or code segments to perform the necessary tasks. A code segment may represent a procedure, a function, a subprogram, a program, a routine, a subroutine, a module, a software package, a class, or any combination of instructions, data structures, or program statements. A code segment may be coupled to another code segment or a hardware circuit by passing and/ or receiving information, data, arguments, parameters, or memory contents. Information, arguments, parameters, data, etc. may be passed, forwarded, or transmitted via any suitable means including memory sharing, message passing, token passing, network transmission, etc. The program or code segments may be stored in a processor readable medium or transmitted by a computer data signal embodied in a carrier wave, or a signal modulated by a carrier, over a transmission medium. The "processor readable medium" may include any medium that can store or transfer information. Examples of the processor readable medium include an electronic circuit, a semiconductor memory

device, a ROM, a flash memory, an erasable ROM (EROM), a floppy diskette, a compact disk (CD-ROM), an optical disk, a hard disk, a fiber optic medium, a radio frequency (RF) link, etc. The computer data signal may include any signal that can propagate over a transmission medium such as electronic network channels, optical fibers, air, electromagnetic, RF links, etc. The code segments may be downloaded via computer networks such as the Internet, Intranet, etc.

It is noted that the invention may be described as a process which is usually depicted as a flowchart, a flow diagram, a structure diagram, or a block diagram. Although a flowchart may describe the operations as a sequential process, many of the operations can be performed in parallel or concurrently. In addition, the order of the operations may be re-arranged. A process is terminated when its operations are completed. A process may correspond to a method, a function, a procedure, a subroutine, a subprogram, etc. When a process corresponds to a function, its termination corresponds to a return of the function to the calling function or the main function.

Figure 1 is a diagram illustrating a system 100 in which one embodiment of the invention can be practiced. The system 100 includes an adaptive filter 110 and a tap-leakage generator 120.

The adaptive filter 110 receives an input signal and generates an output sequence. The adaptive filter 110 may be used in a number of applications such as adaptive equalization in a digital communications systems. The adaptive filter 110 is typically implemented as a non-recursive filter having N taps. An adaptive equalizer can be either symbol-spaced or fractionally spaced. A fractionally spaced equalizer has tap weights spaced a fraction of a symbol internal apart. The fractionally spaced equalizer provides better performance than the conventional symbol-spaced equalizer in the presence of severe linear distortion. The adaptive filter 110 is described here as a symbol-spaced equalizer, but the invention is valid for both symbol and fractionally spaced filters. The adaptive filter 110 has N equalizer coefficients that are updated or adjusted according to the input signal. The adaptive filter 110 prevents the buildup of large coefficient values by systematically adjusting the magnitudes of all the equalizer tap weights or equalizer

coefficients. The adaptive filter 110 generates N equalizer data and a decision error to the tap-leakage generator 120.

The tap-leakage generator 120 updates the N equalizer coefficients used by the adaptive filter 110 using the N equalizer data and the decision error provided by the adaptive filter 110. The tap-leakage generator 120 uses an efficient procedure that employs less computations than the traditional update algorithm with tap-leakage.

Figure 2 is a diagram illustrating the adaptive filter 110 shown in Figure 1 according to one embodiment of the invention. The adaptive filter 110 includes a sampler 210, N delay elements 220_0 to 220_{N-1}, N multipliers 230_0 to 230_{N-1}, an adder 240, a thresholder 250, and an adder 260.

The sampler 210 samples the input signal r(t) into a discrete-time sequence of input data. The sampling frequency is in accordance to the frequency of the input signal meeting the sampling Nyquist criteria. In one embodiment, this sampling frequency is at an integer multiple of the symbol rate corresponding to modem applications. The delay elements 220_0 to 220_{N-1} each delays the input sequence by one sample time period. The sample time period is the sampling time interval of the sampler 210. The delay elements 220_0 to 220_{N-1} may be implemented by a shift register, a number of flip-flops connected in cascade, or any other suitable elements. When implemented by software, the delay elements 220_0 to 220_{N-1} may be implemented an array of elements. Each of the delay elements generates an equalizer data R_n. For N delay elements, there are N equalizer data R_n^0 to R_n^{N-1}.

Each of the multipliers 230_0 to 230_{N-1} multiplies one of the equalizer coefficients D_n^0 to D_n^{N-1} provided by the tap-leakage generator 120 with the corresponding equalizer data. For example, the multiplier 230_0 multiplies the equalizer coefficient D_n^0 by the equalizer data R_n^0, D_n^k with the equalizer data R_n^k, etc. The N equalizer coefficients D_n^0 to D_n^{N-1} are provided by the tap-leakage generator 120.

The adder 240 adds all the products generated by the multipliers to provide a sum S_n. The thresholder 250 thresholds the sum S_n to provide the output sequence A_n. The

adder 260 adds the sum S_n and the output sequence to provide a decision error E_n at each sampling time.

Figure 3 is a diagram illustrating the tap-leakage generator 120 shown in Figure 1 according to one embodiment of the invention. The tap-leakage generator 120 includes an error filter 310 and an updater 320.

The error filter 310 filters the decision error E_n from the adaptive filter 110 using a leakage factor α. The error filter 310 generates a filtered decision error h_n at each sampling time.

The updater 320 updates the vector of N equalizer coefficients D_n^0 to D_n^{N-1} to the N taps using the filtered decision error h_n. The updater receives the N equalizer data from the N taps of the adaptive filter 110. The updater 320 includes N coefficient updaters 330_0 to 330_{N-1}. Each of the N coefficient updaters 330_0 to 330_{N-1} generates a corresponding updated equalizer coefficient. For example, the coefficient updater 330k generates the equalizer coefficient D_n^k. Each of the N coefficient updaters 330_0 to 330_{N-1} uses a corresponding equalizer data. All of the N coefficient updaters 330_0 to 330_{N-1} use a same adaptive step size parameter μ and the filtered decision error h_n.

In the conventional tap-leakage procedure, the updated equalizer coefficients are determined by the following equation:

$$\underline{\mathbf{C}}_{n+1} = \alpha * \underline{\mathbf{C}}_n - \mu * E_n * \underline{\mathbf{R}}_n \qquad (1)$$

where:

$\underline{\mathbf{C}}_n$ is the vector of equalizer coefficients at time n, $\underline{\mathbf{C}}_n^T = [C_n^0 \; C_n^1 \; \ldots \; C_n^{N-1}]$ where T denote vector transposition.

$\underline{\mathbf{R}}_n$ is the vector of equalizer data at time n, $\underline{\mathbf{R}}_n^T = [R_n^0 \; R_n^1 \; \ldots \; R_n^{N-1}]$

E_n is the decision error at time n

μ is the adaptive step size parameter

α is the leakage factor, $0.0 < \alpha < 1.0$.

$*$ is the multiplication operation.

The conventional tap leakage procedure includes the multiplication of α in each of the update path for the equalizer coefficients. The z-transform of the update path can be written as:

$$\underline{\mathbf{C}}_n * z^1 = \alpha * \underline{\mathbf{C}}_n - \mu * E_n * \underline{\mathbf{R}}_n \quad (2)$$

or:

$$\underline{\mathbf{C}}_n = -\mu * E_n * \underline{\mathbf{R}}_n z^{-1} / (1 - \alpha * z^{-1}) \quad (3)$$

The updater 320 removes the multiplication of α from each of the update paths. In addition, the error filter 310 is used to incorporate the leakage factor α. For the tap-leakage generator 120, the equalizer coefficients are now defined as $\underline{\mathbf{D}}_n$ where $\underline{\mathbf{D}}_n^T = [D_n^0 \; D_n^1 \ldots D_n^{N-1}]$. The least mean square (LMS) tap update for the leakage generator 120 is written as:

$$\underline{\mathbf{D}}_{n+1} = \underline{\mathbf{D}}_n - \mu * h_n * \underline{\mathbf{R}}_n \quad (4)$$

where h_n is the filtered decision error.

The Z- transform of the filtered decision error, h_n is:

$$h_n = E_n * h(z) \quad (5)$$

Taking the z-transform of equation (4) to obtain:

$$\mathbf{D}_n * z^1 = \alpha * \mathbf{D}_n - \mu * E_n * H(z) * \mathbf{R}_n \quad (6)$$

or:

$$\mathbf{D}_n = \frac{-\mu * E_n \mathbf{R}_n z^{-1}}{1 - z^{-1}} * H(z) \quad (7)$$

Setting the coefficients $\mathbf{C}_n$ equal to $\mathbf{D}_n$ and solving for H(z) to obtain:

$$H(z) = \left(\frac{1 - z^{-1}}{1 - \alpha * z^{-1}} \right) \quad (8)$$

The block H(z) is the error filter 310 which is a first order high pass filter (HPF) with a zero at zero frequency or direct current (DC) frequency and a pole determined by the leakage factor α.

For $\mathbf{C}_n$ to be identically equal to $\mathbf{D}_n$, the transfer function from E_n to $\mathbf{C}_n$ (and also E_n to $\mathbf{D}_n$) needs to be linear and time-variant. This is not truly the case because $\mathbf{R}_n$ is not

limited to a constant value. However, since the LMS algorithm stochastically estimates the gradient of the data-error product, only the time-average of the data-error product is of concern, and being strictly linear, time-invariant is not required.

The updater 320 therefore eliminates N multiplications due to α and the computation for the error filter 310 is reduced to only three computations. This represents a significant saving when N is large as is typical in adaptive equalization.

Each of the coefficient updater 330k includes a first multiplier 342k, a second multiplier 344k, an adder 346k and a delay element 348k. The first multiplier 342k multiplies a corresponding one of the equalizer data R_n^k with the filtered decision error h_n to provide a first product. The second multiplier 344k multiplies the first product with the adaptive step size μ to provide a second product. The delay element 348k delays the updated equalizer coefficient D_n^k to provide a delayed coefficient. The subtractor 346k subtracts the second product from the delayed coefficient to provide the updated filtered coefficient D_n^k. As seen, the multiplication of the leakage factor α is eliminated. The computation of the updated equalizer coefficient for tap j is given by the following equation:

$$D_n^j = D_{n-1}^j - \mu * h_n * R_n^i \qquad (9)$$

Figure 4 is a diagram illustrating the error filter 310 shown in Figure 3 according to one embodiment of the invention. The error filter 310 is essentially a first-order high pass filter and can be implemented by a number of methods. In one embodiment, the error filter 310 includes first and second computing elements 410 and 420.

The error filter 310 filters the decision error E_n to generate the filtered decision error h_n. The filtered decision error h_n is used by all the coefficient updaters 330_0 to 330_{N-1} as shown in Figure 3. The difference equation for the error filter 310 can be obtained by taking the inverse Z- transform of equation (9):

$$h_n = E_n - E_{n-1} + \alpha * h_{n-1} \qquad (10)$$

where E_{n-1} and h_{n-1} are the delayed versions of the decision error E_n and the filtered decision error h_n, respectively.

The first computing element 410 computes the difference $d_n = E_n - E_{n-1}$ and the second computing element 420 computes $h_n = d_n + \alpha * h_{n-1}$.

The first computing element 410 includes a delay element 412 and a subtractor 414. The delay element 412 delays the decision error E_n by one sample to generate the delayed decision error E_{n-1}. The subtractor 414 subtracts the delayed decision error E_{n-1} from the decision error E_n to provide the error difference d_n.

The second computing element 420 includes an adder 422, a delay element 424, and a multiplier 426. The delay element 424 delays the filtered decision error h_n to provide a delayed output h_{n-1}. The multiplier 426 multiplies the leakage factor with the delayed output to generate a product $P = \alpha * h_{n-1}$. The adder 422 adds the error difference d_n to the product P to generate the filtered decision error h_n.

Figure 5 is a flowchart illustrating a process 500 for adaptive filtering with tap leakage using error filtering according to one embodiment of the invention. It is noted that the START may be located anywhere in the process 500.

Upon START, the process 500 generates equalizer data $\underline{\mathbf{R}}_n$ using the N tap delay elements (Block 510). For N taps, the vector of equalizer data $\underline{\mathbf{R}}_n$ includes R_n^0 to R_n^{N-1}. The generation of the equalizer data R_n may be performed by shifting the elements one sample while acquiring the new sample. Next, the process 500 computes the sum $S_n = \sum R_n^i * \mathbf{D}_n^i$, where i = 0, . . ., N-1 (Block 520). Then, the process 500 thresholds the sum S_n to generate the output sequence A_n (Block 530). Next, the process 500 computes the decision error $E_n = A_n + S_n$ (Block 540). Then, the process 500 filters the decision error E_n by computing the filtered decision error $h_n = E_n - E_{n-1} + \alpha * h_{n-1}$ in accordance to equation (10) (Block 550). The filtered decision error h_n will be used to update the equalizer coefficients D_n.

Next, the process 500 updates the equalizer coefficients $\underline{\mathbf{D}}_n$ using the filtered decision error h_n (Block 560). The updated equalizer coefficients D_n are computed according to equation (9).

Next, the process 500 determines if the termination condition has been met (Block 570). This termination condition may be based on a maximum number of iterations, a maximum time period, or when the system stops working. If so, the process 500 is terminated. Otherwise, the process 500 returns to Block 510.

While this invention has been described with reference to illustrative embodiments, this description is not intended to be construed in a limiting sense. Various modifications of the illustrative embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the spirit and scope of the invention.